Probabilistic Datalog+/- under the Distribution Semantics

Fabrizio Riguzzi, Elena Bellodi, and Evelina Lamma

ENDIF – University of Ferrara, Via Saragat 1, I-44122, Ferrara, Italy
{fabrizio.riguzzi,elena.bellodi,evelina.lamma}@unife.it

Abstract. We apply the distribution semantics for probabilistic ontologies (named DISPONTE) to the Datalog+/- language. In DISPONTE the formulas of a probabilistic ontology can be annotated with an epistemic or a statistical probability. The epistemic probability represents a degree of confidence in the formula, while the statistical probability considers the populations to which the formula is applied. The probability of a query is defined in terms of finite set of finite explanations for the query, where an explanation is a set of possibly instantiated formulas that is sufficient for entailing the query. The probability of a query is computed from the set of explanations by making them mutually exclusive.
We also compare the DISPONTE approach for Datalog+/- ontologies with that of Probabilistic Datalog+/-, where an ontology is composed of a Datalog+/- theory whose formulas are associated to an assignment of values for the random variables of a companion Markov Logic Network.

1 Introduction

Many authors recognize that representing uncertain information is important for the Semantic Web [18, 12] and recently this was also the topic for a series of workshops [6]. Ontologies are a fundamental component of the Semantic Web and Description Logics (DLs) are often the languages of choice for modeling ontologies. Lately much work has focused on developing tractable DLs, such as the *DL-Lite* family [5], for which answering conjunctive queries is in AC_0 in data complexity.

In a related research direction Calì et al. [3] proposed Datalog+/-, a variant of Datalog for defining ontologies. Datalog+/- is able to express the languages of the *DL-Lite* family [2]. Probabilistic Datalog+/- [9, 8] has been proposed for representing uncertainty in Datalog+/-. In this approach an ontology is composed of a Datalog+/- theory and a Markov Logic Network (MLN) [15] and each Datalog+/- formula is associated to an assignment of values to (a subset of) the random variables that are modeled by the MLN. This assignment, called *scenario*, controls the activation of the formulas: they hold only in worlds where the scenario is satisfied.

In the field of logic programming, the distribution semantics [17] has emerged as one of the most effective approaches for integrating logic and probability and underlies many languages such as PRISM [17], ICL [14], Logic Programs with

Annotated Disjunctions [19] and ProbLog [7]. In this semantics the clauses of a probabilistic logic program contain alternative choices annotated with probabilities. Each grounding of a probabilistic clause represents a random variable that can assume a value from the finite set of alternatives. In order to compute the probability of a query, its explanations have to be found, where an explanation is a set of choices that ensure the entailment of the query. The set of explanations must be covering, i.e., it must represent all possible ways of entailing the query. The probability is computed from a covering set of explanations by solving a disjoint sum problem, either using an iterative splitting algorithm [14] or Binary Decision Diagrams [11, 16].

In this paper we apply the distribution semantics to ontological languages and, in particular, to Datalog+/-. We call the approach DISPONTE for "DIstribution Semantics for Probabilistic ONTologiEs" (Spanish for "get ready"). The idea is to annotate formulas of a theory with a probability. We consider two types of probabilistic annotation, an epistemic type, that represents a degree of belief in the formula as a whole, and a statistical type, that considers the populations to which the formula is applied. While in the first case the choice is whether to include or not a formula in an explanation, in the latter case the choice is whether to include instantiations of the formula for specific individuals. The probability of a query is again computed from a covering set of explanations by solving the disjoint sum problem.

The paper is organized as follows. Section 2 provides some preliminaries on Datalog+/-. Section 3 presents DISPONTE while Section 4 describes related work. Section 5 concludes the paper.

2 Datalog+/-

Let us assume (i) an infinite set of data constants Δ, (ii) an infinite set of labeled nulls Δ_N (used as "fresh" Skolem terms) and (iii) an infinite set of variables Δ_V. Different constants represent different values (unique name assumption), while different nulls may represent the same value. We assume a lexicographic order on $\Delta \cup \Delta_N$, with every symbol in Δ_N following all symbols in Δ. We denote by $\mathbf{X}$ vectors of variables $X_1, \ldots, X_k$ with $k \geq 0$. A relational schema $\mathcal{R}$ is a finite set of relation names (or predicates). A term t is a constant, null or variable. An atomic formula (or atom) has the form $p(t_1, \ldots, t_n)$, where p is an n-ary predicate and $t_1, \ldots, t_n$ are terms. A database D for $\mathcal{R}$ is a possibly infinite set of atoms with predicates from $\mathcal{R}$ and arguments from $\Delta \cup \Delta_N$. A conjunctive query (CQ) over $\mathcal{R}$ has the form $q(\mathbf{X}) = \exists \mathbf{Y} \Phi(\mathbf{X}, \mathbf{Y})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms having as arguments variables $\mathbf{X}$ and $\mathbf{Y}$ and constants (but no nulls). A Boolean CQ (BCQ) over $\mathcal{R}$ is a CQ having head predicate q of arity 0 (i.e., no variables in $\mathbf{X}$).

We often write a BCQ omitting the quantifiers. Answers to CQs and BCQs are defined via homomorphisms, which are mappings $\mu : \Delta \cup \Delta_N \cup \Delta_V \rightarrow \Delta \cup \Delta_N \cup \Delta_V$ such that (i) $c \in \Delta$ implies $\mu(c) = c$, (ii) $c \in \Delta_N$ implies $\mu(c) \in \Delta \cup \Delta_N$, and (iii) μ is naturally extended to term vectors, atoms, sets of atoms, and

conjunctions of atoms. The set of all answers to a CQ $q(\mathbf{X}) = \exists \mathbf{Y} \Phi(\mathbf{X}, \mathbf{Y})$ over a database D, denoted $q(D)$, is the set of all tuples $\mathbf{t}$ over Δ for which there exists a homomorphism $\mu : \mathbf{X} \cup \mathbf{Y} \to \Delta \cup \Delta_N$ such that $\mu(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq D$ and $\mu(\mathbf{X}) = \mathbf{t}$. The answer to a BCQ q over a database D is Yes, denoted $D \models q$, iff $q(D) \neq \emptyset$.

A *tuple-generating dependency* (or TGD) F is a first-order formula of the form $\forall \mathbf{X} \forall \mathbf{Y} \Phi(\mathbf{X}, \mathbf{Y}) \to \exists \mathbf{Z} \Psi(\mathbf{X}, \mathbf{Z})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ and $\Psi(\mathbf{X}, \mathbf{Z})$ are conjunctions of atoms over $\mathcal{R}$, called the *body* and the *head* of F, respectively. Such F is satisfied in a database D for $\mathcal{R}$ iff, whenever there exists a homomorphism h such that $h(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq D$, there exists an extension h' of h such that $h'(\Psi(\mathbf{X}, \mathbf{Z})) \subseteq D$. We usually omit the universal quantifiers in TGDs. A TGD is *guarded* iff it contains an atom in its body that involves all variables appearing in the body.

Query answering under TGDs is defined as follows. For a set of TGDs T on $\mathcal{R}$ and a database D for $\mathcal{R}$, the set of models of D given T, denoted $mods(D, T)$, is the set of all (possibly infinite) databases B such that $D \subseteq B$ and every $F \in T$ is satisfied in B. The set of answers to a CQ q on D given T, denoted $ans(q, D, T)$, is the set of all tuples $\mathbf{t}$ such that $\mathbf{t} \in q(B)$ for all $B \in mods(D, T)$. The answer to a BCQ q over D given T is Yes, denoted $D \cup T \models q$, iff $B \models q$ for all $B \in mods(D, T)$.

A Datalog+/- theory may contain also *negative constraints* (or NC), which are first-order formulas of the form $\forall \mathbf{X} \Phi(\mathbf{X}) \to \bot$, where $\Phi(\mathbf{X})$ is a conjunction of atoms (not necessarily guarded). The universal quantifiers are usually left implicit.

Equality-generating dependencies (or EGDs) are the third component of a Datalog+/- theory. An EGD F is a first-order formula of the form $\forall \mathbf{X} \Phi(\mathbf{X}) \to X_i = X_j$, where $\Phi(\mathbf{X})$, called the *body* of F and denoted $body(F)$, is a conjunction of atoms, and X_i and X_j are variables from $\mathbf{X}$. We call $X_i = X_j$ the *head* of F, denoted $head(F)$. Such F is satisfied in a database D for $\mathcal{R}$ iff, whenever there exists a homomorphism h such that $h(\Phi(\mathbf{X})) \subseteq D$, it holds that $h(X_i) = h(X_j)$. We usually omit the universal quantifiers in EGDs. An EGD F on $\mathcal{R}$ of the form $\Phi(\mathbf{X}) \to X_i = X_j$ is applicable to a database D for $\mathcal{R}$ iff there exists a homomorphism $\eta : \Phi(\mathbf{X}) \to D$ such that $\eta(X_i)$ and $\eta(X_j)$ are different and not both constants. If $\eta(X_i)$ and $\eta(X_j)$ are different constants in Δ, then there is a *hard violation* of F. Otherwise, the result of the application of F to D is the database $h(D)$ obtained from D by replacing every occurrence of a non-constant element $e \in \{\eta(X_i), \eta(X_j)\}$ in D by the other element e' (if e and e' are both nulls, then e precedes e' in the lexicographic order).

Example 1. Let us consider the following ontology for a real estate information extraction system, a slight modification of the one presented in [9]:

$F_1 = ann(X, label), ann(X, price), visible(X) \to priceElem(X)$

If X is annotated as a label, as a price, and is visible, then it is a price element.

$F_2 = ann(X, label), ann(X, priceRange), visible(X) \to priceElem(X)$

If X is annotated as a label, as a price range, and is visible, then it is a price element.

$F_3 = priceElem(E), group(E, X) \to forSale(X)$

If E is a price element and is grouped with X, then X is for sale.

$F_4 = forSale(X) \rightarrow \exists Pprice(X, P)$

If X is for sale, then there exists a price for X.

$F_5 = hasCode(X, C), codeLoc(C, L) \rightarrow loc(X, L)$

If X has postal code C, and C's location is L, then X's location is L.

$F_6 = hasCode(X, C) \rightarrow \exists L codeLoc(C, L), loc(X, L)$

If X has postal code C, then there exists L such that C has location L and so does X.

$F_7 = loc(X, L1), loc(X, L2) \rightarrow L1 = L2$

If X has the locations $L1$ and $L2$, then $L1$ and $L2$ are the same.

$F_8 = loc(X, L) \rightarrow advertised(X)$

If X has a location L then X is advertised.

Suppose we are given the database

$$codeLoc(ox1, central), codeLoc(ox1, south), codeLoc(ox2, summertown)$$
$$hasCode(prop1, ox2), ann(e1, price), ann(e1, label), visible(e1),$$
$$group(e1, prop1)$$

The atomic BCQs $priceElem(e1)$, $forSale(prop1)$ and $advertised(prop1)$ evaluate to true, while the CQ $loc(prop1, L)$ has answers $q(L) = \{summertown\}$. In fact, even if $loc(prop1, z_1)$ with $z_1 \in \Delta_N$ is entailed by formula F_5, formula F_7 imposes that $summertown = z_1$. If F_7 were absent, then $q(L) = \{summertown, z_1\}$.

The *chase* is a bottom-up procedure for repairing a database relative to a Datalog+/- theory and can be used for deriving atoms entailed by the database and the theory. If such a theory contains only TGDs, the chase consists of an exhaustive application of the TGD chase rule in a breadth-first fashion. The TGD chase rule consists in adding to the database the head of a TGD if there is an homomorphism between the body and the current database. In order to fill the arguments of the head occupied by existentially quantified variables, "fresh" null values are used.

A BCQ can be answered by performing the chase and checking whether the query is entailed by the extended database that is obtained.

Answering BCQs q over databases, guarded TGDs and NC can be done by, for each constraint $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow \bot$, checking that the BCQ $\Phi(\mathbf{X})$ evaluates to false; if one of these checks fails, then the answer to the original BCQ q is positive, otherwise the negative constraints can be simply ignored when answering the original BCQ q.

The chase in the presence of both TGDs and EGDs is computed by iteratively applying (1) a single TGD once and (2) the EGDs, as long as they are applicable (i.e., until a fix point is reached). EGDs are assumed to be separable [4]. Intuitively, separability holds whenever: (i) if there is a hard violation of an EGD in the chase, then there is also one on the database w.r.t. the set of EGDs alone (i.e., without considering the TGDs); and (ii) if there is no hard violation,

then the answers to a BCQ w.r.t. the entire set of dependencies equals those w.r.t. the TGDs alone (i.e., without the EGDs).

A guarded Datalog+/- ontology consists of a database D, a finite set of guarded TGDs T_T, a finite set of negative constraints T_C and a finite set of EGDs T_E that are separable from T_T. The data complexity (i.e., the complexity where both the query and the theory are fixed) of evaluating BCQs relative to a guarded Datalog+/- theory is polynomial [1].

3 The DISPONTE Semantics for Probabilistic Ontologies

A *probabilistic ontology* (D, T) consists of a database D and a set T of *certain formulas*, that take the form of a Datalog+/- TGD, NC or EGD, of *epistemic probabilistic formulas* of the form

$$p_i ::_e F_i \tag{1}$$

where p_i is a real number in $[0, 1]$ and F_i is a TGD, NC or EGD, and of *statistical probabilistic formulas* of the form

$$p_i ::_s F_i \tag{2}$$

where p_i is a real number in $[0, 1]$ and F_i is a TGD.

In formulas of the form (1), p_i is interpreted as an epistemic probability, i.e., as the degree of our belief in formula F_i, while in formulas of the form (2), p_i is interpreted as a statistical probability, i.e., as information regarding random individuals from certain populations. These two types of statements can be related to the work of Halpern [10]: an epistemic statement is a Type 2 statement and a statistical statement is a Type 1 statement.

For example, an epistemic probabilistic concept inclusion TGD of the form

$$p ::_e c(X) \rightarrow d(X) \tag{3}$$

represents the fact that we believe in the truth of $c \subseteq d$, where c and d are interpreted as sets of individuals, with probability p. A statistical probabilistic concept inclusion TGD of the form

$$p ::_s c(X) \rightarrow d(X) \tag{4}$$

instead means that a random individual of class c has probability p of belonging to d, thus representing the statistical information that a fraction p of the individuals of c belongs to d. In this way, the overlap between c and d is quantified. The difference between the two formulas is that, if two individuals belong to class c, the probability that they both belong to d according to (3) is p while according to (4) is $p \times p$.

The idea of DISPONTE is to associate independent Boolean random variables to (instantiations of) the formulas. By assigning values to every random variable we obtain a *world*, the set of logic formulas whose random variable is

assigned to 1. Note that the assumption of independence of the random variables does not limit the set of distributions over the ground logical atoms that can be represented: by possibly introducing extra atoms, any distribution over the atoms that can be represented with a Bayesian network can be represented with a probabilistic ontology.

To clarify what we mean by instantiations, we now define substitutions. Given a formula F, a *substitution* θ is a set of couples X/x where X is a variable universally quantified in the outermost quantifier in F and $x \in \Delta \cup \Delta_N$. The application of θ to F, indicated by $F\theta$, is obtained by replacing X with x in F and by removing X from the external quantification for every couple X/x in θ. An *instantiation* of a formula F is the result of applying a substitution to F.

To obtain a world w of a probabilistic ontology T, we include every certain formula in w. For each axiom of the form (1), we decide whether or not to include it in w. For each axiom of the form (2), we generate all the substitutions for the variables universally quantified in the outermost quantifier and for each instantiation we decide whether or not to include it in w.

There may be an infinite number of instantiations. For each instantiated formula, we decide whether or not to include it in w. In this way we obtain a Datalog+/- theory which can be assigned a semantics as seen in Section 2.

To formally define the semantics of a probabilistic ontology we follow the approach of Poole [14]. An *atomic choice* in this context is a triple (F_i, θ_j, k) where F_i is the i-th formula, θ_j is a substitution and $k \in \{0, 1\}$. If F_i is obtained from a certain formula, then $\theta_j = \emptyset$ and $k = 1$. If F_i is obtained from a formula of the form (1), then $\theta_j = \emptyset$. If F_i is obtained from a formula of the form (2), then θ_j instantiates the variables universally quantified in the outermost quantifier.

A *composite choice* κ is a consistent set of atomic choices, i.e., $(F_i, \theta_j, k) \in \kappa, (F_i, \theta_j, m) \in \kappa \Rightarrow k = m$ (only one decision for each formula). The probability of composite choice κ is $P(\kappa) = \prod_{(F_i,\theta_j,1)\in\kappa} p_i \prod_{(F_i,\theta_j,0)\in\kappa}(1 - p_i)$. A *selection* σ is a total composite choice, i.e., it contains one atomic choice (F_i, θ_j, k) for every instantiation $F_i\theta_j$ of formulas of the theory. Since the domain is infinite, every selection is, too. Let us indicate with $\mathcal{S}_T$ the set of all selections. $\mathcal{S}_T$ is infinite as well. A selection σ identifies a theory w_σ called a *world* in this way: $w_\sigma = \{F_i\theta_j | (F_i, \theta_j, 1) \in \sigma\}$. Let us indicate with $\mathcal{W}_T$ the set of all worlds. A composite choice κ identifies a set of worlds $\omega_\kappa = \{w_\sigma | \sigma \in \mathcal{S}_T, \sigma \supseteq \kappa\}$. We define the set of worlds identified by a set of composite choices K as $\omega_K = \bigcup_{\kappa\in K} \omega_\kappa$.

A composite choice κ is an *explanation* for a BCG query q if q is entailed by the database and every world of ω_κ. A set of composite choices K is *covering* with respect to q if every world w_σ in which q is entailed is such that $w_\sigma \in \omega_K$. Two composite choices κ_1 and κ_2 are *incompatible* if their union is inconsistent. A set K of composite choices is *mutually incompatible* if for all $\kappa_1 \in K, \kappa_2 \in K, \kappa_1 \neq \kappa_2 \Rightarrow \kappa_1$ and κ_2 are incompatible.

Explanations can be found by keeping track of the formulas that were used for adding atoms to the database in the chase procedure.

Kolmogorov defined probability functions (or measures) as real-valued functions over an algebra Ω of subsets of a set $\mathcal{W}$ called the *sample space*. The

set Ω is an algebra of $\mathcal{W}$ iff (1) $\mathcal{W} \in \Omega$, (2) Ω is closed under complementation, i.e., $\omega \in \Omega \rightarrow (\mathcal{W} \setminus \omega) \in \Omega$ and (3) Ω is closed under finite union, i.e., $\omega_1 \in \Omega, \omega_2 \in \Omega \rightarrow (\omega_1 \cup \omega_2) \in \Omega$. The elements of Ω are called *measurable sets*. Not every subset of $\mathcal{W}$ need be present in Ω.

Given a sample space $\mathcal{W}$ and an algebra Ω of subsets of $\mathcal{W}$, a probability measure is a function $\mu : \Omega \rightarrow R$ that satisfies the following axioms: (1) $\mu(\omega) \geq 0$ for all $\omega \in \Omega$, (2) $\mu(\mathcal{W}) = 1$, (3) $\omega_1 \cap \omega_2 = \emptyset \rightarrow \mu(\omega_1 \cup \omega_2) = \mu(\omega_1) + \mu(\omega_2)$ for all $\omega_1 \in \Omega, \omega_2 \in \Omega$.

Poole [14] proposed an algorithm, called *splitting algorithm*, to obtain a set of mutually incompatible K' composite choices from any set of composite choices K such that $\omega_K = \omega_{K'}$. Moreover, he proved that if K_1 and K_2 are both mutually incompatible finite sets of finite composite choices such that $\omega_{K_1} = \omega_{K_2}$ then $\sum_{\kappa \in K_1} P(\kappa) = \sum_{\kappa \in K_2} P(\kappa)$.

These results also hold for the probabilistic ontologies we consider, so we can define a unique probability measure $\mu : \Omega_T \rightarrow [0, 1]$ where Ω_T is defined as the set of sets of worlds identified by finite sets of finite composite choices: $\Omega_T = \{\omega_K | K$ is a finite set of finite composite choices$\}$. It is easy to see that Ω_T is an algebra over $\mathcal{W}_T$.

Then μ is defined by $\mu(\omega_K) = \sum_{\kappa \in K'} P(\kappa)$ where K' is a finite mutually incompatible set of finite composite choices such that $\omega_K = \omega_{K'}$. $\langle \mathcal{W}_T, \Omega_T, \mu \rangle$ is a probability space according to Kolmogorov's definition.

The probability of a BCQ query q is given by $P(q) = \mu(\{w | w \in \mathcal{W}_T \wedge D \cup w \models q\})$. If q has a finite set K of finite explanations such that K is covering then $\{w | w \in \mathcal{W}_T \wedge D \cup w \models q\} \in \Omega_T$ and $P(q)$ is well-defined.

Example 2. Let us consider the following probabilistic ontology, obtained from the one presented in Example 1 by adding probabilistic annotations:

$$
\begin{aligned}
0.4 ::_s\ & F_1 = ann(X, label), ann(X, price), visible(X) \rightarrow priceElem(X) \\
0.5 ::_s\ & F_2 = ann(X, label), ann(X, priceRange), visible(X) \rightarrow priceElem(X) \\
0.6 ::_s\ & F_3 = priceElem(E), group(E, X) \rightarrow forSale(X) \\
& F_4 = forSale(X) \rightarrow \exists P price(X, P) \\
& F_5 = hasCode(X, C), codeLoc(C, L) \rightarrow loc(X, L) \\
& F_6 = hasCode(X, C) \rightarrow \exists L codeLoc(C, L), loc(X, L) \\
0.8 ::_e\ & F_7 = loc(X, L1), loc(X, L2) \rightarrow L1 = L2 \\
0.7 ::_s\ & F_8 = loc(X, L) \rightarrow advertised(X)
\end{aligned}
$$

and the database of Example 1:

$$
\begin{aligned}
& codeLoc(ox1, central), codeLoc(ox1, south), codeLoc(ox2, summertown), \\
& hasCode(prop1, ox2), ann(e1, price), ann(e1, label), visible(e1), \\
& group(e1, prop1)
\end{aligned}
$$

A covering set of explanations for the query $q = priceElem(e1)$ is $K = \{\kappa_1\}$ where $\kappa_1 = \{(F_1, \{X/e1\}, 1)\}$. K is also mutually exclusive so $P(q) = 0.4$.

A covering set of explanations for the query $q = forSale(prop1)$ is $K = \{\kappa_1, \kappa_2\}$ where $\kappa_1 = \{(F_1, \{X/prop1\}, 1), (F_3, \{X/prop1\}, 1)\}$ and $\kappa_2 = \{(F_2, \{X/prop1\}, 1), (F_3, \{X/prop1\}, 1)\}$.

An equivalent mutually exclusive set of explanations obtained by applying the splitting algorithm is $K' = \{\kappa'_1, \kappa'_2\}$ where $\kappa'_1 = \{(F_1, \{X/prop1\}, 1), (F_3, \{X/prop1\}, 1), (F_2, \{X/prop1\}, 0)\}$ and $\kappa'_2 = \{(F_2, \{X/prop1\}, 1),$
$(F_3, \{X/prop1\}\ , 1)\}$ so $P(q) = 0.4 \cdot 0.6 \cdot 0.5 + 0.5 \cdot 0.6 = 0.42$.

A covering set of explanations for the query $q = advertised(prop1)$ is $K = \{\kappa_1, \kappa_2, \kappa_3\}$ with

$$\kappa_1 = \{(F_8, \{X/prop1, L/summertown\}, 1), (F_7, \emptyset, 1)\}$$
$$\kappa_2 = \{(F_8, \{X/prop1, L/summertown\}, 1), (F_7, \emptyset, 0)\}$$
$$\kappa_3 = \{(F_8, \{X/prop1, L/z_1\}, 1), (F_7, \emptyset, 0)\}$$

where $z_1 \in \Delta_N$. A mutually exclusive set of explanations is $K' = \{\kappa'_1, \kappa'_2, \kappa'_3\}$ where

$$\kappa'_1 = \{(F_8, \{X/prop1, L/summertown\}, 1), (F_7, \emptyset, 1)\}$$
$$\kappa'_2 = \{(F_8, \{X/prop1, L/summertown\}, 1), (F_7, \emptyset, 0), (F_8, \{X/prop1, L/z_1\}, 0)\}$$
$$\kappa'_3 = \{(F_8, \{X/prop1, L/z_1\}, 1), (F_7, \emptyset, 0)\}$$

so $P(q) = 0.7 \cdot 0.8 + 0.7 \cdot 0.2 \cdot 0.3 + 0.7 \cdot 0.2 = 0.742$

Example 3. Let us consider the following ontology, inspired by the `people+pets` ontology proposed in Patel-Schneider et al. [13]:

$$0.5 ::_s F_1 = hasAnimal(X, Y), pet(Y) \rightarrow petOwner(X)$$
$$0.6 ::_s F_2 = cat(X) \rightarrow pet(X)$$

and the database $hasAnimal(kevin, fluffy), hasAnimal(kevin, tom), cat(fluffy), cat(tom)$. A covering set of explanations for the query $q = petOwner(kevin)$ is $K = \{\kappa_1, \kappa_2\}$ where $\kappa_1 = \{(F_1, \{X/kevin\}, 1), (F_2, \{X/fluffy\}, 1)\}$ and $\kappa_2 = \{(F_1, \{X/kevin\}, 1), (F_2, \{X/tom\}, 1)\}$. An equivalent mutually exclusive set of explanations is $K' = \{\kappa'_1, \kappa'_2\}$ where:

$$\kappa'_1 = \{(F_1, \{X/kevin\}, 1), (F_2, \{X/fluffy\}, 1), (F_2, \{X/tom\}, 0)\}$$
$$\kappa'_2 = \{(F_1, \{X/kevin\}, 1), (F_2, \{X/tom\}, 1)\}$$

so $P(q) = 0.5 \cdot 0.6 \cdot 0.4 + 0.5 \cdot 0.6 = 0.42$

Example 4. Let us consider the following ontology:

$$F_1 = \exists Y hasAnimal(X, Y), pet(Y) \rightarrow petOwner(X)$$
$$0.6 ::_s F_2 = cat(X) \rightarrow pet(X)$$
$$0.4 ::_e F_3 = cat(fluffy)$$
$$0.3 ::_e F_4 = cat(tom)$$

and the database $hasAnimal(kevin, fluffy), hasAnimal(kevin, tom)$. A covering set of explanations for the query axiom $q = petOwner(kevin)$ is $K = \{\kappa_1, \kappa_2\}$ where

$$\kappa_1 = \{(F_3, \emptyset, 1), (F_2, \{X/fluffy\}, 1)\}$$
$$\kappa_2 = \{(F_4, \emptyset, 1), (F_2, \{X/tom\}, 1)\}$$

which, after splitting, becomes $K' = \{\kappa'_1, \kappa'_2, \kappa'_3\}$:

$$\kappa'_1 = \{(F_3, \emptyset, 1), (F_2, \{X/fluffy\}, 1), (F_4, \emptyset, 1), (F_2, \{X/tom\}, 0)\}$$
$$\kappa'_2 = \{(F_3, \emptyset, 1), (F_2, \{X/fluffy\}, 1), (F_4, \emptyset, 0)\}$$
$$\kappa'_3 = \{(F_4, \emptyset, 1), (F_2, \{X/tom\}, 1)\}$$

so $P(q) = 0.4 \cdot 0.6 \cdot 0.3 \cdot 0.4 + 0.4 \cdot 0.6 \cdot 0.7 + 0.3 \cdot 0.6 = 0.3768$

4 Related Work

Gottlob et al. [9, 8] present probabilistic Datalog+/-, a version of Datalog+/- that allows the representation of probabilistic information by combining Markov Logic Networks with Datalog+/-. Each Datalog+/- formula F is annotated with a *probabilistic scenario* λ, an assignment of values to a set of random variables from the MLN associated to the ontology. A full probabilistic scenario assigns a value to all the random variables of the MLN. A probabilistic scenario represents an event that happens when the random variables described by the MLN assume the values indicate in the scenario. Probabilistic formulas then take the form $F : \lambda$.

A probabilistic Datalog+/- is of the form $\Phi = (O, M)$ where O is a set of annotated formulas and M is a MLN. An annotated formula holds when the event associated with its probabilistic annotation holds.

If a is a ground atom, its probability in a probabilistic Datalog+/- ontology $\Phi = (O, M)$, denoted $Pr(a)$, is obtained by summing the probabilities according to M of all full scenarios such that the atom is entailed by the annotated formulas that hold in the scenario.

Example 5. Let us consider the following probabilistic Datalog+/- ontology from [8]:

$$F_1 = visible(X) \rightarrow priceElem(X) : \{ann(X, label), ann(X, price)\}$$
$$F_2 = visible(X) \rightarrow priceElem(X) : \{ann(X, label), ann(X, priceRange)\}$$
$$F_3 = priceElem(E), group(E, X) \rightarrow forSale(X) : \{sale\}$$
$$F_4 = forSale(E) \rightarrow \exists P price(X, P)$$
$$F_5 = hasCode(X, C), codeLoc(C, L) \rightarrow loc(X, L)$$
$$F_6 = hasCode(X, C) \rightarrow \exists L codeLoc(C, L), loc(X, L)$$
$$F_7 = loc(X, L1), loc(X, L2) \rightarrow L1 = L2 : \{uniqueLoc\}$$

and the MLN

$$0.3 \;\; ann(X, label) \wedge ann(X, price)$$
$$0.4 \;\; ann(X, label) \wedge ann(X, priceRange)$$
$$0.8 \;\; sale$$
$$1.1 \;\; uniqueLoc$$

Suppose that this network is grounded with respect to the only constant $e1$. The resulting ground network has 5 Boolean random variables, each corresponding to a logical atom. Therefore, there are 2^5 full scenarios. In this theory $Pr(priceElem(e1)) = 0.492$ and $Pr(forSale(prop1)) = 0.339$.

5 Conclusions

We have presented the application of the distribution semantics for probabilistic ontologies (named DISPONTE) to the Datalog+/- language. DISPONTE is inspired by the distribution semantics of probabilistic logic programming and is a minimal extension of the underlying ontology semantics to allow to represent and reason with uncertain knowledge.

DISPONTE differs from Probabilistic Datalog+/- because the probabilistic interactions among the atoms are modeled directly by means of Datalog+/- formulas rather than by a separate entity. The parameters of DISPONTE Datalog+/- are easier to interpret as they are probabilities (statistical or epistemic) while MLN parameters are weights not directly interpretable as probabilities. Moreover, DISPONTE does not require the prior grounding of the probabilistic atoms, for which the set of constants has to be defined by the user, but allows an on demand grounding on the basis of the terms that are used for inference.

In the future we plan to design inference algorithms for probabilistic Datalog+/- under the DISPONTE semantics.

References

1. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. In: International Conference on Principles of Knowledge Representation and Reasoning. pp. 70–80. AAAI Press (2008)
2. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. In: Symposium on Principles of Database Systems. pp. 77–86. ACM (2009)
3. Calì, A., Gottlob, G., Lukasiewicz, T.: Tractable query answering over ontologies with Datalog+/-. In: International Workshop on Description Logics. CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009)
4. Calì, A., Gottlob, G., Lukasiewicz, T., Marnette, B., Pieris, A.: Datalog+/-: A family of logical knowledge representation and query languages for new applications. In: IEEE Symposium on Logic in Computer Science. pp. 228–242. IEEE Computer Society (2010)

5. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. Autom. Reasoning 39(3), 385–429 (2007)
6. Costa, P.C.G., d'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.): Uncertainty Reasoning for the Semantic Web I, ISWC International Workshops, URSW 2005-2007, Revised Selected and Invited Papers, LNCS, vol. 5327. Springer (2008)
7. De Raedt, L., Kimmig, A., Toivonen, H.: ProbLog: A probabilistic Prolog and its application in link discovery. In: International Joint Conference on Artificial Intelligence. pp. 2462–2467 (2007)
8. Gottlob, G., Lukasiewicz, T., Simari, G.I.: Answering threshold queries in probabilistic Datalog+/- ontologies. In: International Conference on Scalable Uncertainty Management. LNCS, vol. 6929, pp. 401–414. Springer (2011)
9. Gottlob, G., Lukasiewicz, T., Simari, G.I.: Conjunctive query answering in probabilistic Datalog+/- ontologies. In: International Conference on Web Reasoning and Rule Systems. LNCS, vol. 6902, pp. 77–92. Springer (2011)
10. Halpern, J.Y.: An analysis of first-order logics of probability. Artif. Intell. 46(3), 311–350 (1990)
11. Kimmig, A., Demoen, B., Raedt, L.D., Costa, V.S., Rocha, R.: On the implementation of the probabilistic logic programming language ProbLog. Theor. Prac. Log. Prog. 11(2-3), 235–262 (2011)
12. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the semantic web. J. Web Sem. 6(4), 291–308 (2008)
13. Patel-Schneider, P, F., Horrocks, I., Bechhofer, S.: Tutorial on OWL. In: International Semantic Web Conference (2003), `http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial/`
14. Poole, D.: Abducing through negation as failure: stable models within the independent choice logic. J. Log. Prog. 44(1-3), 5–35 (2000)
15. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. 62(1-2), 107–136 (2006)
16. Riguzzi, F.: Extended semantics and inference for the Independent Choice Logic. Log. J. IGPL 17(6), 589–629 (2009)
17. Sato, T.: A statistical learning method for logic programs with distribution semantics. In: International Conference on Logic Programming. pp. 715–729. MIT Press (1995)
18. URW3-XG: Uncertainty reasoning for the World Wide Web, final report (2005)
19. Vennekens, J., Verbaeten, S., Bruynooghe, M.: Logic programs with annotated disjunctions. In: International Conference on Logic Programming. LNCS, vol. 3131, pp. 195–209. Springer (2004)